

# SPECIFICATION

Docket No. 0544MH-35309

TO ALL WHOM IT MAY CONCERN:

BE IT KNOWN that we, Ajit Sagar, Israel Hilerio, residing in the state of Texas, and Vijayasathy S. Chakravarthy; residing in the State of California, have invented new and useful improvements in an

## INTERNET COMMUNICATIONS METHOD

of which the following is a specification:

## BACKGROUND OF THE INVENTION

### 1. Field of the Invention:

The present invention relates generally to communications systems and control for computer networks, and more specifically to an intelligent control system for use with a message exchange network.

### 2. Description of the Prior Art:

The recent dramatic increases in communications bandwidth and capability are enabling businesses to work using techniques not previously available or contemplated. The high level of communication available between companies allow them to provide on-line, real-time manufacturing, ordering, and shipping control capabilities. The communications networks currently coming into place will allow companies to enter orders, receive responses, and perform other manufacturing and shipping related tasks as if two companies were directly linked and closely related.

High-level communications interconnectivity between companies allows them to establish relationships not hereto possible. Separate companies are beginning to establish communication links which allow them to cooperate much more closely with suppliers and customers. For example, systems currently coming into use are allowing companies to take an order for a customer, confirm availability of products from suppliers, and



[illegible]

2

3

4

5

6

7

8

## BRIEF DESCRIPTION OF THE DRAWINGS

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself however, as well as a preferred mode of use, further objects and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

Figure 1 is a high level diagram showing an exchange communication system for transacting business between numerous companies;

Figure 2 is a dataflow diagram illustrating an exchange of messages between companies;

Figure 3 is a block diagram showing the structure of a preferred event-condition-action control system;

Figures 4 and 5 are block diagrams illustrating examples of how the preferred embodiment functions;

Figures 6, 7, and 8 are petri-net examples illustrating operation of control logic of the preferred embodiment; and

1 Figure 9 is a petri-net illustrating control flow in a transaction example.

Figure 9 is a petri-net illustrating control flow in a transaction example.

## DESCRIPTION OF THE PREFERRED EMBODIMENT

1

2           In the preferred embodiment, a centralized communications service, hereinafter  
3 referred to as an exchange 10, is provided in communication with numerous corporate  
4 computer systems. As shown in Figure 1, users of the exchange are grouped into suppliers  
5 12, manufacturers 14, resellers 16, customers 18, and logistics 20. It will be understood by  
6 those skilled in the art that each of these areas is represented by numerous companies. In  
7 addition, any one company may fall under different categories at different times. For  
8 example, a manufacturer may have numerous suppliers, each of which considers that  
9 manufacturer to be a customer. Each of the suppliers may in turn have suppliers of their  
10 own. Companies designated as resellers may be considered as suppliers to one  
11 manufacturer or customer, and customers to another. It will be understood that the  
12 functional groupings shown in Figure 1 are for convenience only, and many relationships  
13 are not easily formed into a simplistic definition.

14

15           As contemplated by the present invention, the exchange service provides a  
16 mechanism for routing messages between companies. These messages can be formatted in  
17 numerous ways so that companies having disparate computer systems can communicate  
18 effectively. Preferably, the messaging system is independent of the system designs used by  
19 various companies, but the particular messaging system utilized does not itself form a part  
20 of the present invention.

21

1 As referred to in Figure 1, those companies designated as logistics are generally  
2 shippers of physical items. Including shipment details in a communications network so that  
3 they can be accessed improves efficiency of the overall system.  
4

5 Referring to Figure 2, an example is shown of a particular simple transaction to  
6 illustrate how the exchange works. In this extremely simple example, four separate  
7 companies are designated as customer 22, reseller 24, manufacturer 26, and logistics 28, or  
8 transportation. As described above, the relationships between these companies can change  
9 in the context of a different transaction.  
10

11 In the transaction shown in Figure 2, the customer 22 sends an order message A  
12 through the exchange 10 to reseller 24. This order can be a firm order, a request for a  
13 quotation, or similar request. In order to determine whether the order can be filled, and  
14 various terms such as shipping date, reseller 24 sends messages B and C to manufacturer  
15 26 and logistics 28 respectively. Messages B and C pass through the exchange 10 to these  
16 companies. The manufacturer 26 determines the terms upon which it can supply the order,  
17 and returns message D through the exchange 10 to reseller 24. At the same time, the  
18 logistics company 28 determines availability of shipping, and returns message E with this  
19 information to reseller 24. In some transactions, shipping information may pass between  
20 manufacturer 26 and logistics company 28, with shipping being a part of message D  
21 returned from manufacturer 26 to reseller 24.  
22



1 In the present example, reseller 24 determines availability in terms of the order  
2 based upon a promise from the manufacturer 26. In addition, shipping date and terms are  
3 determined. This information is placed into message F, which is returned through the  
4 exchange 10 to customer 22.

5  
6 In this example, placing an order and the relationships between various companies  
7 are straightforward. However, use of the exchange 10 becomes more valuable if it can  
8 contain intelligence of its own, and perform more complex tasks.

9  
10 For example, with access to many suppliers, customers will often want to request  
11 quotes from several suppliers, or possibly even select one or more suppliers through an  
12 auction or similar process. This can involve the customer placing an order to a shared  
13 location in a manner that is available to all interested suppliers. Any supplier who wishes to  
14 bid on the order can do so, and the exchange can handle collecting quotes and making it  
15 available to the customer as is described below.

16  
17 In a similar manner, suppliers can automate, or partially automate, the interface  
18 between themselves and their potential customers. When orders are placed, simple orders  
19 can be responded to automatically. For example, in the same communications sequence  
20 given above in Figure 2, messages B and C generated by reseller need not be generated as  
21 the result of human interaction. Instead, if this order is one of a standard type which fits  
22 certain parameters, as selected by reseller 24, messages B and C can be generated  
23 automatically upon receipt of a qualifying order. This type of automated message handling

1 can, of course, be provided independently by the reseller, but in the preferred embodiment  
2 of the present invention certain functions are available within the exchange itself. This  
3 provides enhanced flexibility and service to companies using the exchange, with minimum  
4 software generation requirements for these companies.

5  
6 The exchange itself is depicted in the drawings as a single, central object. However,  
7 in reality it will be a multi-part, highly distributed service. Insofar as the various users are  
8 concerned, the exchange will look like a single object in the same manner that most users  
9 view the Internet today. However, the various pieces of the exchange will be located on a  
10 large number of systems, providing capacity, flexibility, and system robustness. Preferably,  
11 backup devices and fault tolerant systems are provided using techniques known in the art.

12  
13 Referring to Figure 3, a logical structure for message handling within the exchange  
14 is shown. Not all messages passed through the exchange will need to be handled in this  
15 manner; however, in the preferred embodiment most or all messages are treated in the same  
16 way to simplify the design. Preferably, all messages are treated as events, as will now be  
17 described.

18  
19 Conceptionally, handling of messages within the exchange is broken into three parts.  
20 An event container 30 accepts incoming events (messages) 32, and stores date and time  
21 information about them. As described below, the event container preferably contains a  
22 timer 34, enabling time sensitive events to be handled. A condition container 36 contains

1 instances of conditions 38, which are generally supplied by users of the system. An action  
2 container 40 contains instances of actions 42, which are also generally supplied by users.

3

4 When a user desires the exchange to perform an intelligent response or filtering, a  
5 message is sent. For example, if a customer wishes to obtain goods to supply an order, a  
6 request for quotes message can be sent to the appropriate companies, or posted to a central  
7 location made available in the exchange. Replies to the message, which will consist of  
8 quotations by various suppliers, will be accepted as events by the exchange and handled in a  
9 manner designated by the center of the original message. All messages 32 come into the  
10 event container 30, and are stored there for further processing.

11

12 The timer 34 is used to generate events related to the clock or the calendar. If, for  
13 example, the customer wants to consider only bids which are submitted within a particular  
14 time window, responsive messages are time stamped and compared with timing events  
15 generated by the timer 34.

16

17 Within the condition container 36 are numerous instances of conditions 38 which  
18 have been defined by users of the exchange. In the example of a customer putting an order  
19 out for bids, conditions regarding receipt of those bids, for example, can be defined as  
20 condition instances. Any type of condition desired by the customer can be implemented in  
21 the conditions instances. These are implemented as logical relationships between  
22 characteristics of the events, such as time, number, and value of various parameters. For  
23 example, the customer could want to consider only the first three responsive events, or only

1 responses returned before close of business on the same day as the request, and so forth.  
2 These types of logical conditions are expressed in the condition instances as is described  
3 below in more detail.  
4

5 The action container 40 contains instances of actions 42 which are to occur in  
6 response to conditions being met. Typically, the actions will be to generate additional  
7 events. In such case, actions which occur are also returned as events to the event container  
8 30.

9  
10 Breaking the function of the exchange into these three conceptual blocks allows  
11 many changes to be made dynamically. For example, changes can be made to conditions  
12 without affecting events which have already occurred. Because events are stored in the  
13 event container, conditions can be modified as desired by the user without impacting the  
14 event container. As is described further below, once a condition, whether original or  
15 modified, has been met, the events fulfilling that condition are removed from the event  
16 container 30.

17  
18 In a similar manner, actions can be changed independently of events or conditions.  
19 When a condition instance is changed, it will be common to change the corresponding  
20 action instance. However, these two sets of instances are not tightly tied together, and may  
21 be modified independently.  
22

1 Each container utilizes a listener to watch for incoming events. This will typically  
2 be interrupt driven, so that something will be done within the container when the listener  
3 detects that an event has arrived. Within the event container, the events are stored and  
4 catalogued. Additionally, conditional determinations may be made as described below. In  
5 the condition container, when an event arrives from the event container, the condition  
6 framework, or engine, determines which conditions may be affected by the event. There  
7 may be more than one such condition. The framework then determines whether any of the  
8 potentially affected conditions are satisfied, and if so an event is sent to the action container.

9  
10 Within the action container, receipt of an event by the listener causes the appropriate  
11 action or actions to be performed. As described above, some of these actions will be the  
12 generation of an event which is returned to the event container, where they are detected by  
13 the event listener.

14  
15 Figure 4 is an example illustrating how the exchange functions in a simple instance.  
16 In this example, a business can automatically accept simple orders within certain  
17 parameters, and send a return message to the customer promising fulfillment of the order. In  
18 the example of Figure 4, orders are to be processed only between 8:00 a.m. and 6:00 p.m.  
19 This rule, referred to as an Event-Condition-Action (ECA) rule, is set up to deal with a  
20 certain specific case. Other ECA rules would be setup for other ordering conditions.

21  
22 Referring to Figure 4, an event framework so is an operational portion of the event  
23 container. It contains a listener, which constantly scans for events. The timer 34 generates

1 timing events, 52, 54 which are recognized by the event framework 50. In other words, the  
2 event framework is aware of the current time. In the present case, incoming events are only  
3 to be processed between the hours of 8:00 a.m. and 6:00 p.m.  
4

5 In the preferred embodiment, the event framework contains conditions in addition to  
6 those contained in the condition container. Conditions in the event container are preferably  
7 a small subset of possible conditions, directed to timing and counting of events. Thus, a rule  
8 within the event framework can provide that a message is sent to the condition container  
9 only if an order is received between the timed events of 8:00 a.m. and 6:00 p.m. Another  
10 type of condition preferably implemented in the event framework is an event counting  
11 condition, such as "take an action once three proposals have been received." Such counting  
12 of events is preferably a task performed within the event container. If desired, all of these  
13 timing and counting conditions could be implemented in the condition container, but in a  
14 large system, the condition container will generally contain many complex conditions set up  
15 by system users. Low level decisions, such as time related or count related decisions, can  
16 easily be implemented within the event container without adding to the complexity already  
17 inherent in the large number of conditions in the condition container.  
18

19 Event2 56 is an order by a customer which has a quantity of 400, and a price of  
20 \$4,500. The listener of the event framework 50 recognizes the occurrence of Event2, and  
21 determines that it occurs between Event1 (8:00 a.m.) and Event3 (6:00 p.m.). It therefore  
22 generates an Event2', containing the terms of the order, and sent to the condition container.

1 If, as described above, the time related conditions are instead implemented in the condition  
2 container, the just described condition would be processed there.

3

4 The condition instance 58 in the condition container, set up by the company  
5 accepting orders, specifies that this condition is triggered if an order comes in having a  
6 quantity less than 500, and a price less than \$5000. Once the listener within the condition  
7 framework notices the occurrence of Event2', which meets these conditions, Event2'' is  
8 generated. Event2'' is an order having the previously noted quantity and price. Event2'' is  
9 sent to the action instance 60, which defines the actions to be taken when such an order is  
10 received. Once the listener within the action container notes the occurrence of Event2'', a  
11 promise to fulfill the order is obtained and the order is sent to the company for processing.  
12 Event4 62 is generated, which is a return promise back to the customer. The customer will  
13 presumably provide its own conditions for handling a promise such as Event4, but the  
14 message may simply be forwarded by the system to be handled by a person in the usual  
15 way.

16

17 Referring to Figure 5, the same situation is illustrated, except that two separate  
18 actions are connected to this condition instance. In addition to the action 62 described in  
19 connection with Figure 4, an additional action 64 is provided which logs the order to a  
20 database 66 so that it is available to the company. Any number of actions may be attached  
21 to a single condition instance.

22





1

2           Conceptually, the internal logic within the condition container and the event  
3 container utilizes the concept of “petri-nets”. As is known in the art, this is a conceptual  
4 framework which allows for generation of actions in response to asynchronous events, and  
5 persistence in the manner described above. Simple examples of petri-nets are shown in  
6 Figures 6-8, and will be recognized by those familiar with this technology.

7

8           Referring to Figure 6, a simple petri-net which corresponds to the conjunction of two  
9 events generating an action is shown. The first and second event, represented by circles  
10 and 72, correspond to “places” in petri-net terminology. Action event 74 also corresponds  
11 to a place. The condition instance 76 corresponds to a transition. In Figure 6, the transition  
12 occurs if both the first and second events have occurred, causing the resulting action 74 to be  
13 generated. The first two places correspond to events received by the event container, and  
14 the resulting place corresponds to an event generated by an action instance.

15

16           Figure 7 shows a similar petri-net diagram, with the first and second events 78, 80  
17 being combined in a logical or operation. If either event occurs, the resulting action event  
18 82 is generated.

19

20           Figure 8 shows an event that is a composite of composite events. As will be  
21 appreciated by those skilled in the art, nets-nets can be logically combined to any level of  
22 complexity to define the desired condition. Figure 8 shows a petri-net for (E1 or (E2 and

1 E3)). In other words, an E4, corresponding to an event generated by an action instance, is  
2 generated when either E1 or both of E2 and E3 occur.

3

4 Manipulation of petri-nets calls for tokens to be placed in various places. When all  
5 of the places which provide an input to a transition are filled, these tokens are all removed  
6 and tokens are placed in all output places. This corresponds conceptually to the generation  
7 of persistent events in the event container, followed by removal of these events and  
8 generation of action events as described above. A transition corresponds to a condition  
9 instance, and output places correspond to actions. Conceptually, a petri-net separates inputs  
10 from outputs, in a manner similar to separation of ECA events into the three separate event,  
11 condition and action containers.

12

13 Figure 9 is a more complex petri-net representing a condition similar to the request  
14 for three quotes described above. In this set of conditions, the customer desires to make a  
15 selection only when three separate quotes have been submitted in response to a request.  
16 When each of the quotes Q1, Q2, and Q3 have been submitted, a transition occurs which  
17 generates two outputs 82, 84. The first output action 82 is an acknowledgement to all who  
18 have submitted quotes that the quotes have been received, and the second output action 84 is  
19 submission of the quotes to a selection process. This may be automated, or may be reported  
20 to a person to make decision as to which quote is to be accepted. If selection is automated,  
21 the selection may be as complex as necessary. The selector action 84 represents activity  
22 which may take place out of the exchange, by sending appropriate messages to the company  
23 which will be returned when a decision has been made. Once a decision has been made and





1           While the invention has been particularly shown and described with reference to a  
2   preferred embodiment, it will be understood by those skilled in the art that various changes  
3   in form and detail may be made therein without departing from the spirit and scope of the  
4   invention.

5

[illegible]